Süsteemprogrammeerimine keeles C

Loeng 1

milles tutvume C kursuse korraldusega, saame aimu kodutöödest, küsime endalt miks me siin kursusel oleme, saame teada millest kursus seekord räägib ning vaatleme keele põhielemente ja C-s kirjutatud näiteprogramme

Tänane menüü

- Korralduslik info
- Kodutööd ja tähtajad
- Sellest, millest kursus räägib
- C keele triviaalne osa koos selles tehtavate vigadega

Sellest, millest kursus räägib

- Süsteemprogrammeerimine keeles C õpetab C keele kasutamist Posix (Unix/Linux/Misiganux) keskkonnas.
  - *Selline esialgne selgitus - varsti veidi põhjalikumalt*

Korralduslik info

- Läbiviimine:
  - Jaagup Irve, 56463800, jaagup.irve@ttu.ee
- Veebileht:
  - http://www.lambda.ee/ -> Süsteemprogrammeerimine keeles C
  - http://www.lambda.ee/index.php/itv0020

Ajad

- Loengud
  - Esmaspäeval kell 16:00 X-212
- Praktikumid
  - ruum IT-213E, I (võib-olla muutub)
  - Teisipäeviti
  - 12:00 IAPB53
  - 14:00 IAPB51, IAPB53

Õhtuseks lugemiseks

- Põhiõpikud:
  - Brian W. Kernighan, Dennis M. Ritchie *The C Programming Language, Second Edition*, Pretince Hall 1988
  - Randal E. Bryant and David R. O'Hallaron *Computer Systems: A Programmer's Perspective (CS:APP)*, Prentice Hall, 2003

Eeldame:

- Programmeerimiskogemus
- Mõningane kogemus UNIX keskkonnaga
- Kasutajakonto praktikumiklassis
- Tahtejõud vähemalt ühe labori valmis kirjutamiseks

Harivad kodutööd

- 2 kodutööd
  - Faili pakkimine (keeruline , aga hariv)
  - Primitiivne FTP (vähekeeruline, aga tüütu)
- 20% hindest (+lisapunktid)
- Kaitsmine praktikumide ajal
  - Programmide kopeerimine/ümberkirjutamine tähendab aine uut deklareerimist
- *Tähelepanek: Kodutööde valmiskirjutamine õpetab möödaminnes ka eksami selgeks.*

Miks pole tänavu tööka üliõpilase alternatiivi?

- Sest teie eelkäijad kirjutasid nad kõik maha
- Selle asemel püüan tundidesse harjutusi välja mõelda

Põnev eksam

- 7 ülesannet
- Palju aega
- Teooria ja praktilised ülesanded
- Kaks teooriaküsimust (triviaalne, keerulisem)
- Kolm programmeerimisülesannet (triviaalne, tavaline ja tüütu)
- Kaks nuputusülesannet (lihtne ja keeruline)
- Programmeerimine paberile

Miks me siin kursusel oleme?

Ainepunktid?
vs.
Siiras huvi C keele vastu?

Järgneb õppejõu meeleheitlik katse teid viimase poole ümber veenda

C keele omadused

- Mahult väike
- Paindlikud andmetüübid
  - tüübivigu eriti ei ole - ise tead, mis teed
- Pointerid – kasutatakse mälu ja massiivide tarbeks
- Struktuurid, funktsioonid
- Võimaldab madalatasemelist bitiväänamist

Plusspool

- Levinud ja populaarne
- Porditav (ANSI C)
- C-s kirjutatud programmid on tõhusad (samas ei ole ka teised keeled ilmtingimata halvemad)
- Tüübivabadus (int – char)
- Hea riistvara programmeerimiseks
- C on abiks teiste keelte tundmisel

Miinuspool

- C on keeruline – tegelikult ei ole
- C programmikood pole arusaadav – programmeerija teha
- C-s on võimalik kergesti vigu teha - kasvate inimesena
- Puudub sisseehitatud mäluhaldus (vt eelmine)
- Puudub objektorienteeritus – kompenseerimiseks loodi nt C++ ja Objective-C dialektid.

Kursuse ülevaade

- C keele süntaktiline osa
- C keele kompileerimise etapid
- Sisend-väljund, failid, stream'id
- Mäluhaldus, pointerid
- Failimajandus - deskriptorid, kataloogid, atribuudid
- Võrgundus - kliendi ja serveri pool
- Protsesside haldamine ja signaalid
- Lõpus kordamist ja detaile
- Võtmesõna: käsurea programmid

Hello, World!

- C programm „Tere, Maailm“:

```
#include <stdio.h>
main() {
   printf(„Hello, World!\n“);
}
```

Hello, World!

- C programm „Tere, Maailm“:

```
#include <stdio.h>
main() {
   printf(„Hello, World!\n“);
}
```

- Kaasab faili stdio.h. Võimaldab printf kasutamise

Hello, World!

- C programm „Tere, Maailm“:

```
#include <stdio.h>
main() {
   printf(„Hello, World!\n“);
}
```

- Funktsioon *main()*, millest alustatakse kõikide C programmide täitmist

Hello, World

- C programm „Tere, Maailm“:

```
#include <stdio.h>
main() {
   printf(„Hello, World!\n“);
}
```

- Looksulud defineerivad koodibloki, antud juhul funktsiooni *main()* sisu

Hello, World

- C programm „Tere, Maailm“:

```
#include <stdio.h>
main() {
   printf(„Hello, World!\n“);
}
```

- Funktsiooni *printf()* väljakutse. Väljastab argumendina toodud teksti.
- \n – reavahetus (*newline*)

Keerulisem näide

```
typedef unsigned char B;char*x[]={
#include "dict.h"
0};typedef struct L{B*s;struct L*n;}L;
L*h[128],*l[128],*s[128],Z[sizeof x/sizeof*x],*F=Z;int c[256],m,a=1;
int k(B*q){int g=0;B*p=q;while(*p)g|=!c[*p++]--;return g-1&p-q;}
void u(B*p){while(*p)c[*p++]++;}
void S(int N,int r,int t,L*W){L*w;int i,n;
 for(n=r<N?r:N;n>0;n--)for(w=n==N?W:h[n];s[t]=w;u(w->s),w=w->n)if(k(w->s))
  if(n==r){if(t==m-1)for(i=a=0;i<=t;i++)printf("%s%c",s[i]->s,i<t?' ':'\n');}
  else if(t<m-1)S(n,r-n,t+1,s[t]=w);}
int main(int C,B**A){int i=0,g,n=0;B*p;while(--C)for(p=*++A;n<127&&*p;)c[*p++]++,n+
   +;
 for(;p=x[i++];u(p))if(g=k(p))(l[g]=*(l[g]?&l[g]->n:&h[g])=F++)->s=p;
 while(++m<128)S(127,n,0,h[127]);
 return a;}
```

- Peter Klausler, IOCCC 2006 (http://www.ioccc.org/)

Ajalugu

- Kuulub ALGOLil põhinevasse keelterühma
- ALGOL60->CPL->BCPL->B->C
- C: 1972 by Ken Thompson & Dennis Ritchie, Bell Labs (AT&T); DEC PTP-11
- 1986 C++ Bjarne Stroustrup
- 1989 ANSI C (C89)
- 1990 ISO standard (C90 = C98)
- 1999 ISO standard (C99)

C võtmesõnad

- Muutujate tüübid
  - `char double enum float int long short struct union void`
- Muutujate parameetrid
  - `auto const extern register signed static unsigned volatile`
- Programmivoo kontroll
  - `break case continue default do else for goto if return switch while`
- Operaatorid
  - `sizeof`

Kompileerijast

- Kasutame GNU C kompilaatorit gcc
- Kasutamisjuhised esimeses praktikumis

Filosoofiline arutelu kursuse olemusest

Muutujad

- Nimega mäluaadress
- Omavad tüüpi, millest sõltub tõlgendamine, suurus
- Tuleb enne kasutamist deklareerida (tema omadused tuleb kompileerijale teatavaks teha)
- Näidisdeklaratsioonid:

```
int i;              /* i is an integer */
float f;            /* f is a floating point number */
char c1, c2;        /* c1 and c2 are characters */
```

Primitiivsed andmetüübid

- Andmetüüp – määrab kuidas andmeid tõlgendada
- Andmetüübid:
  - `int` (täisarvud)
  - `float` (ujukomaarvud)
  - `double` (topeltpikkusega ujukomaarv)
  - `char` (tähemärk)
- **NB!** String ei ole C-s primitiivne tüüp!
- Andmete maht võib erinevate C implementatsioonide vahel erineda. Andmetüübi tegeliku mahu baitides saame `sizeof` operaatori abil.
  - `sizeof(char) = 1`
  - `sizeof(int) = 4`
  - `sizeof(float) = 4`
  - `sizeof(double) = 8`

Täisarvud

- *Integer* jaguneb:

```
int j;  /* one bit used for sign, the rest for value */
unsigned int k;     /* all bits used for value – larger
  value can be stored */
```

- Erinevad suurused: `short`, `long`, `long long`
- Tõeväärtused (*boolean*): puudub eraldi tüübina, kasutatakse int-i – 0 = FALSE, kõik muu TRUE (reeglina 1)
- Tehted (operaatorid):

```
int j = 1; /* initialization */
j = j+1    /* increment value, same as j++ */
j--; /* decrement value, same as j = j-1 or j -= 1 */
j *= 6     /* multiply value by 6, same as j = j*6 */
j /= 6     /* divide by 6 *NOTE*: Integer division! */
```

Täisarvu hoidmine

- Mälus bittmustrina
  - 0000 0000 0000 0000 0000 0000 0000 0010 = 2
  - 1111 1111 1111 1111 1111 1111 1111 1110 = -2
- Piirid (16 bitise täisarvu puhul):
  - unsigned: 0..65535
  - signed: -32768..32767
- Bittoperatsioonid `unsigned int` märksõnaga

Expression, Statement

- Expression = operand, operator. Väärtustatakse. Jaguneb:
  - Võrdlustehted: x>y, 2==y, x!=5
  - Aritmeetilised tehted: x+2, y--, j*j, 6/3
  - Omistamistehe: x=y, x=4
  - **NB!** Ära võrdle nii: x=y /* VALE! */
- Statement – funktsioonide komponendid
  - lihtne x=y;
  - liit- {x=5; y=z=3; f* = 5.0; }
  - tsüklid for(..) , while(..)
  - if-laused if, if..else

Tsüklid

- For-tsükkel

```
for (initialization statement; test statement; iteration
   statement)
    statement;
```

Näide:

```
for (j=0; j<5; j++)
    printf("Spargel!\n");
```

Näide:

```
j=0;
for ( ; ; )
{
    printf("Spargel!\n");
    if (j>=5) break;
    j++;
}
```

Tsüklid (2)

- While-tsükkel

```
while (test)
   statement;
```

Näide:

```
j=0;
while ( j<5 )
{
   printf(“Spargel!\n”);
   j++;
}
```

Samaväärne:

```
for (j=0; j<5; j++)
   printf(“Spargel!\n”);
```

Tsüklid (4)

- Do-while-tsükkel

```
do
   statement
while (test);
```

Näide:

```
j=0;
do
{
   printf(“Spargel veelkord!\n”);
   j++;
} while ( j<5 );
```

Samaväärne:

```
for (j=0; j<5; j++)
   printf(“Spargel veelkord!\n”);
```

if

```
if (test)
   statement /* käivitub, kui test pole 0 */
```

Näide:

```
if (x>1)
   x=1;
```

Näide:

```
if (x>100)
{
   x=100;
   printf("X on rohkem kui 100, kärbime ta 100-ks\n");
}
```

Näide:

```
if (x>1) if (x>2) x = 0;
```

if (2)

```
if (test)
   true-statement; /* käivitub, kui test pole 0 */
else
   false-statement; /* käivitub, kui test on 0 */
```

Näide:

```
if (j==k)
   printf(“j on võrdne k-ga\n”);
else if (j<k)
   printf(“j on väiksem kui k\n”);
else
   printf(“j on suurem kui k\n”);
```

Funktsioonid

- Funktsiooni definitsioon:

```
int square(int a) {
   return (a*a);
}
```

- Funktsiooni deklaratsioon:

```
int square(int a); /* prototype - now square can be used*/
```

- Funktsioon tuleb kasutamiseks defineerida
- Funktsioon deklareeritakse kui ta on enne kasutamist defineerimata
- Prototüübis on kirjas kogu info funktsiooni väljakutsuva koodi loomiseks:
  - nimi;
  - tagastatava väärtuse tüüp;
  - argumentide arv ja tüüp;

Funktsioonid (2)

Näide:

```
#include <stdio.h>
int square(int);
main(){
int x,y=2;
x=square(y);                    /*function call*/
printf("square of %d is %d\n", y, x); /* another
  function call*/
}
```

Standardsisend & -väljund

- Standardsisend – klaviatuurilt
- Standardväljund – ekraan
- Standardsed librafunktsioonid:
  - Sisend: scanf(), getchar(), gets()
  - Väljund: printf(), putchar(), puts()
- printf() & scanf() on siintoodutest kõige võimsamad – suutelised töötama erinevate tüüpidega

```
int j; float f; double d; char c;
printf("%d %f %lf %c\n", j,f,d,c);
scanf("%d %f %lf %c\n", &j,&f,&d,&c);
```

- & - address operator - when reading value into a variable we use &.

Sisend-väljund

```
main(){
   int j;
   for (j=0; j<5; j++){
      printf("Hello!\n");
      printf("j=%d\n", j);
   }
}
```

```
Hello!
j=0
Hello!
j=1
Hello!
j=2
Hello!
j=3
Hello!
j=4
```

Näidisprogramm

```
#include <stdio.h>
main() {
   int fahr, celsius;
   int lower, upper, step;
   lower = 0;  /* lower limit of temperature table */
   upper = 300;      /* upper limit of temperature table */
   step  = 20; /* step size */
   fahr = lower;
   while(fahr <= upper) {
      celsius = 5 * (fahr-32) /9;
      printf ("%d\t%d\n", fahr, celsius);
       fahr = fahr + step;
   }
}
```